2016 China Collegiate Programming Contest Final

Problems

Problem A. The Third Cup is Free

Time limit: 3 seconds

Panda and his friends were hiking in the forest. They came across a coffee bar inside a giant tree trunk.

Panda decided to treat everyone a cup of coffee and have some rest. Mr. Buck, the bartender greeted Panda and his animal friends with his antler. He proudly told them that his coffee is the best in the forest and this bar is a Michelin-starred bar, thats why the bar is called Starred Bucks.

There was a campaign running at the coffee bar: for every 3 cups of coffee, the cheapest one is FREE. After asking all his friends for their flavors, Panda wondered how much he need to pay.

Input

The first line of the input gives the number of test cases, T.

T test cases follow. Each test case consists of two lines. The first line contains one integer N, the number of cups to be bought.

The second line contains N integers $p_1, p_2, \cdots, p_N$ representing the prices of each cup of coffee.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from `1`) and `y` is the least amount of money Panda need to pay.

Limits

- $1 \le T \le 100$.
- $1 \le N \le 10^5$.
- $1 \le p_i \le 1000$.

Sample input and output

Sample Input	Sample Output
2 3 1 2 3 5 10 20 30 20 20	Case #1: 5 Case #2: 80

Problem B. Wash

Time limit: 10 seconds

Mr.Panda is about to engage in his favourite activity doing laundry! He's brought L indistinguishable loads of laundry to his local laundromat, which has N washing machines and M dryers. The i^{th} washing machine takes W_i minutes to wash one load of laundry, and the i^{th} dryer takes D_i minutes to dry a load of laundry.

At any point in time, each machine may only be processing at most one load of laundry.

As one might expect, Panda wants to wash and then dry each of his L loads of laundry. Each load of laundry will go through the following steps in order:

1. A non-negative amount of time after Panda arrives at the laundromat, Panda places the load in an unoccupied washing machine i.
2. W_i minutes later, he removes the load from the washing machine, placing it in a temporary holding basket (which has unlimited space)
3. A non-negative amount of time later, he places the load in an unoccupied dryer j
4. D_j minutes later, he removes the load from the dryer

Panda can instantaneously add laundry to or remove laundry from a machine. Help Panda minimize the amount of time (in minutes after he arrives at the laundromat) after which he can be done washing and drying all L loads of laundry!

Input

The first line of the input gives the number of test cases, T.

T test cases follow. Each test case consists of three lines. The first line contains three integer L, N, and M.

The second line contains N integers W_1, W_2, $\cdots$, W_N representing the wash time of each wash machine.

The third line contains M integers D_1, D_2, ..., D_M representing the dry time of each dryer.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from `1`) and `y` is the minimum time it will take Panda to finish his laundry.

Limits

- $1 \leq T \leq 100$.
- $1 \leq L \leq 10^6$.
- $1 \leq N, M \leq 10^5$.
- $1 \leq W_i, D_i \leq 10^9$.

Sample input and output

Sample Input	Sample Output
2 1 1 1 1200 34 2 3 2 100 10 1 10 10	Case #1: 1234 Case #2: 12

Problem C. Mr.Panda and Survey

Time limit: 15 seconds

Mr.Panda did a survey among N candidates.

In the survey, each candidate was given a questionnaire which contains M yes/no questions. It is guaranteed that each question was answered with a "Yes" or a "No" by each candidate.

Mr.Panda likes variety. He considers a question as a good question if there are at least one candidate answered "Yes" and at least one candidate answered "No" for that question.

Now Mr.Panda has collected all the questionnaires. He wants to do some statistical analysis based on the survey result.

Because Mr.Panda is super lazy, he wants to randomly pick some of the questionnaires as a sample.

For each possible subset of questions (excluding empty set), Mr.Panda wants to know the probability that all questions in the subset are good questions, assuming that questionnaires in the sample are chosen randomly so that sample can be any subset (including full set and empty set) of questionnaires with equal probability.

Could you help Mr.Panda solve this problem?

To simplify the problem, you are only required to calculate

$$(P_1 \cdot 2^N \mod 1000000007) \oplus (P_2 \cdot 2^N \mod 1000000007) \oplus \cdots \oplus (P_{2^M-1} \cdot 2^N \mod 1000000007)$$

where P_i means the probability of i^{th} subset of questions to be good questions. It is obvious that $P_i \cdot 2^N$ is always an integer.

"$\oplus$" which is known as "bitwise exclusive or" corresponds to operator "^"in both C/C++ and Java.

" $\mod$ " which is known as "modulo" corresponds to operator "%" in both C/C++ and Java.

Input

The first line of the input gives the number of test cases, T.

T test cases follow. Each test case consists of two lines. The first line contains two integers N, the number of questionnaires, and M, the number of questions.

The second line contains N strings $Q_1, Q_2, \cdots, Q_N$ representing the answers in each questionnaire.

Each questionnaire Q_i is given in the form of exact M charecters where each character can be either "Y" standing for "Yes" or "N" standing for "No".

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1) and y is the xor sum of the weighted probabilities.

Limits

- $1 \le T \le 20$.
- $1 \le N \le 10^5$.
- $1 \le M \le 15$.

Sample input and output

Sample Input	Sample Output
2 2 2 NY YN 4 2 NN NY YN YY	Case #1: 1 Case #2: 7

Note

`Case #1`: The probability of question 1 to be good: $\frac{1}{4}$
The probability of question 2 to be good: $\frac{1}{4}$
The probability of both question 1 and question 2 to be good: $\frac{1}{4}$
Thus the answer is

$$(1 \mod 1000000007) \oplus (1 \mod 1000000007) \oplus (1 \mod 1000000007) = 1$$

`Case #2`: The probability of question 1 to be good: $\frac{9}{16}$
The probability of question 2 to be good: $\frac{9}{16}$
The probability of both question 1 and question 2 to be good: $\frac{7}{16}$
Thus the answer is

$$(9 \mod 1000000007) \oplus (9 \mod 1000000007) \oplus (7 \mod 1000000007) = 7$$

Problem D. Game Leader

Time limit: 2 seconds

Recently Tom is playing an interesting game. The game contains a social network, so players can make friends with each other. The friend relation is symmetric, which means if A is a friend of B, B is also a friend of A.

Players' ratings are distinct from each other, which means there are no two players sharing a same rating. Each player owns a ranklist containing all his friends' rating (including himself) and there is one leader who has the highest rating in each player's ranklist.

Tom has N friends. During the communications with these friends, he knows some pairs of friends are also friends. But he doesn't know all the such pairs. In other words, some friends maybe are friends but Tom doesn't know it.

One day, Tom noticed that the system started showing the number of leaders for every friend. E.g. the system may says Peter is leader on 3 players' ranklist.

Now Tom can see the number of leaders for every friend. He would like to know the least number of strangers' ranklist leader are his friends.

Input

The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with a line consists of 3 integers, N, M and R representing the number of people on Tom's ranklist(including Tom), the number of friendships Tom knows and the rank of Tom in his ranklist(`1`-indexed). The next line consists of N integers, the i^{th} integer C_i represents the number of leaders for the friend on rank i. The following M lines each consists of 2 integers X_i, Y_i, means that the friend on rank X_i is a friend of friend on rank Y_i.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from `1`) and `y` is the least number of ranklists that not owned by Tom's friend but the leader is Tom's friend.

Limits

- $1 \leq T \leq 100$.
- $1 \leq N, M \leq 10^5$.
- $1 \leq X_i \leq N$.
- $1 \leq Y_i \leq N$.
- $1 \leq R \leq N$.
- $R \neq X_i$.
- $R \neq Y_i$.
- $X_i \neq Y_i$.
- $0 \leq C_i \leq 10^9$.

- It's guaranteed that the data is valid.

Sample input and output

Sample Input	Sample Output
2 4 0 3 1 0 0 2 4 1 4 1 0 2 0 1 3	Case #1: 2 Case #2: 2

Problem E. Problem Buyer

Time limit: 1 second

TopSetter is an organization that creates problems. They've prepared N problems with estimated difficulty score in range $[A_i, B_i]$. TopHoster would like to host a contest consisting of M problems. The i^{th} problem should be of difficulty score C_i. The i^{th} problem from TopSetter can be used in the contest if and only if its estimated difficulty score range $[A_i, B_i]$ covers the difficulty score c of its target problem in the contest, i.e. $A_i \le c \le B_i$. Hosting a contest with M problems needs to have M distinct problems which satisfy the required difficulty scores for each problem.

Unfortunately, TopSetter doesn't provide a service to buy specific problems. You can only request a problem set containing K problems and they will give you K distinct problems from all the N problems, but you don't know which problems will be given.

As TopSetter is the only problem provider for TopHoster, TopHoster would like to know the least number K of problems they need to buy to make sure they can host a contest.

Input

The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with 2 integers, N and M. Then N lines follow, each line consists of 2 integers representing the difficulty score range of the i^{th} problem, A_i and B_i. The last line of each test case consists of M integers representing the target difficulty scores of the M problems C_i.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from `1`) and `y` is the least number of problems which the TopHoster needs to buy. Output "`IMPOSSIBLE!`" if it's impossible.

Limits

- $1 \le T \le 100$.
- $1 \le N, M \le 10^5$.
- $1 \le A_i \le B_i \le 10^9$.
- $1 \le C_i \le 10^9$.

Sample input and output

Sample Input	Sample Output
3 3 1 1 4 2 3 5 6 3 3 2 1 10 3 4 7 9 4 8 3 3 1 2 5 6 8 9 1 5 10	Case #1: 2 Case #2: 2 Case #3: IMPOSSIBLE!

Problem F. Periodical Cicadas

Time limit: 5 seconds

Nearly all cicadas spend years underground as juveniles, before emerging above ground for a short adult stage of several weeks to a few months.

The seven periodical cicada species are so named because, in any one location, all of the members of the population are developmentally synchronized they emerge as adults all at once every seven years.

The lifecycles of most periodical cicada species range from two to seventeen years, although some could be longer.

There is a forest which can be roughly divided into a matrix of size $N \times M$. The upper-left region is $(1, 1)$ and the lower-right region is (N, M).

A population of periodical cicadas live within each region of the matrix. The population in region (i, j) emerged in year $a_{i,j}$ for the first time, and re-emerges every $b_{i,j}$ years. i.e. they are $b_{i,j}$ - periodical cicadas.

Given a selected rectangular area, entomologists wonder if there is a time when all cicadas in that area emerge together.

Input

The first line of the input gives the number of test cases, T. T test cases follow. Each test cases begin with two integers N and M.

The following N lines each consists of M integers $a_{i,j}$ representing the year cicadas emerged in region (i, j) for the first time.

The following N more lines each consists of M integers $b_{i,j}$ representing the periodical cycle of the cicadas in that region.

Then comes a line with an integer Q representing the number of queries. The following Q lines each consists of 4 integers: x_1, y_1, x_2, y_2, representing the upper-left and lower-right coordinate of the selected rectangular area.

Output

For each test case, first output one line containing "`Case #x:`", where `x` is the test case number (starting from `1`).

The following Q lines each consists of an integer which is the year when all cicadas in the selected area emerge together for the first time or -1 if it's impossible.

Limits

- $1 \le T \le 10$.
- $1 \le N, M \le 200$.
- $0 \le a_{i,j} < b_{i,j} \le 40$.
- $1 \le x_1 \le x_2 \le N$.
- $1 \le y_1 \le y_2 \le M$.

- $1 \leq Q \leq 500000$.

Sample input and output

<table>
<tr><th>Sample Input</th><th>Sample Output</th></tr>
<tr><td>2
3 4
3 1 1 2
1 1 2 1
1 0 5 5
5 4 2 3
2 2 3 2
4 2 6 6
5
2 2 2 2
1 1 3 4
1 4 2 4
1 1 1 2
2 2 3 4
1 2
0 1
2 2
1
1 1 1 2</td><td>Case #1:
1
-1
5
13
-1
Case #2:
-1</td></tr>
</table>

Problem G. Pandaland

Time limit: 3 seconds

Mr. Panda lives in Pandaland. There are many cities in Pandaland. Each city can be treated as a point on a 2D plane. Different cities are located in different locations.

There are also M bidirectional roads connecting those cities. There is no intersection between two distinct roads except their endpoints. Besides, each road has a cost w.

One day, Mr. Panda wants to find a simple cycle with minmal cost in the Pandaland. To clarify, a simple cycle is a path which starts and ends on the same city and visits each road at most once. The cost of a cycle is the sum of the costs of all the roads it contains.

Input

The first line of the input gives the number of test cases, T. T test cases follow.

Each test case begins with an integer M.

Following M lines discribes roads in Pandaland.

Each line has 5 integers x_1, y_1, x_2, y_2, w, representing there is a road with cost w connecting the cities on (x_1, y_1) and (x_2, y_2).

Output

For each test case, output one line containing `Case #x: y`, where `x` is the test case number (starting from `1`) and `y` is the cost Mr. Panda wants to know.

If there is no cycles in the map, `y` is `0`.

Limits

- $1 \le T \le 50$.
- $1 \le m \le 4000$.
- $-10000 \le x_i, y_i \le 10000$.
- $1 \le w \le 10^5$.

Sample input and output

Sample Input	Sample Output
2 5 0 0 0 1 2 0 0 1 0 2 0 1 1 1 2 1 0 1 1 2 1 0 0 1 5 9 1 1 3 1 1 1 1 1 3 2 3 1 3 3 2 1 3 3 3 1 1 1 2 2 2 2 2 3 3 3 3 1 2 2 1 2 2 1 3 2 4 1 5 1 4	Case #1: 8 Case #2: 4

Problem H. Engineer Assignment

Time limit: 1 second

In Google, there are many experts of different areas. For example, MapReduce experts, Bigtable experts, SQL experts, etc. Directors need to properly assign experts to various projects in order to make the projects going smoothly.

There are N projects owned by a director. For the i^{th} project, it needs C_i different areas of experts, $a_{i,0}$, $a_{i,1}$, $\cdots$, a_{i,C_i-1} respective. There are M engineers reporting to the director. For the i^{th} engineer, he is an expert of D_i different areas, $b_{i,0}$, $b_{i,1}$, $\cdots$, b_{i,D_i-1}.

Each engineer can only be assigned to one project and the director can assign several engineers to a project. A project can only be finished successfully if the engineers expert areas covers the project areas, which means, for each necessary area of the project, there is at least one engineer masters it.

The director wants to know how many projects can be successfully finished.

Input

The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with a line consisting of 2 integers, N the number of projects and M the number of engineers. Then N lines follow. The i^{th} line containing the information of the i^{th} project starts with an integer C_i then C_i integers follow, $a_{i,0}$, $a_{i,1}$, $\cdots$, a_{i,C_i-1} representing the expert areas needed for the i^{th} project. Then another M lines follow. The i^{th} line containing the information of the i^{th} engineer starts with an integer D_i then D_i integers follow, $b_{i,0}$, $b_{i,1}$, $\cdots$, b_{i,D_i-1} representing the expert areas mastered by i^{th} engineer.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from `1`) and `y` is the maximum number of projects can be successfully finished.

Limits

- $1 \le T \le 100$.
- $1 \le N, M \le 10$.
- $1 \le C_i \le 3$.
- $1 \le D_i \le 2$.
- $1 \le a_{i,j}, b_{i,j} \le 100$.

Sample input and output

Sample Input	Sample Output
1 3 4 3 40 77 64 3 10 40 20 3 40 20 77 2 40 77 2 77 64 2 40 10 2 20 77	Case #1: 2

Note

For the first test case, there are 3 projects and 4 engineers. One of the optimal solution is to assign the first(40 77) and second engineer(77 64) to project 1, which could cover the necessary areas 40, 77, 64. Assign the third(40 10) and forth(20 77) engineer to project 2, which could cover the necessary areas 10, 40, 20. There are other solutions, but none of them can finish all 3 projects. So the answer is 2.

Problem I. Mr. Panda and Crystal

Time limit: 2 seconds

Long long time ago, there is a magic continent far far away.

There are N types of magic crystals that contain ancient magic powers. Each of the type of magic crystal has its own price for one piece in the market. As the most powerful magician, Mr. Panda could synthesize some types of crystals by collecting some amount of other types of crystals. He could also create some types of crystals by using some number of his magic powers.

Now, Mr Panda can create any number of crystals as he wish by using no more than M magic powers. He want to know the maximum amount of money he can make by sell all the crytals he creates and synthesizes.

Input

The first line of the input gives the number of test cases, T. T test cases follow.

Each test case starts with 3 positive intergers, M, N and K represent the amount of magic powers Mr. Panda had, the number of crystal types on the magic continent and the number of crystal synthesis equations.

Then N lines follows, each of them starts with one `0` or `1` which indicates whehter Mr. Panda could create this type of crystal.

If the i^{th} line starts with `0`, which means Mr. Panda couldn't create crystal type i. Then there is one integer p_i in this line which is the price for each piece of crystal type i.

If the i^{th} line starts with 1, which means Mr. Panda could create crystal type i. Then there are two positive integers c_i and p_i in this line, the first is the amout of magic power cost when creates one piece of crystal type i, and the second is is the price for each piece of crystal type i.

The following K lines each start with two interger x_i and y_i, which means for synthesizing one piece of crystal type x_i, y_i rules should be satisfied. Then there are y_i pair of positive intergers u_j and v_j means for one piece of x_i^{th} type cristal, we have to collect v_i piece of crystal type u_i. Only when **all** the rules of u_i and v_i are satisfied, Mr. Panda could synthesize one piece x_i^{th} type cristal.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from `1`) and `y` is the maximum amout of money Mr. Panda could make.

Limits

- $1 \leq T \leq 100$.
- $1 \leq M \leq 10000$.
- $1 \leq N \leq 200$.
- $1 \leq K \leq 200$.
- $1 \leq x_i, u_j \leq N$.
- for each crystal synthesis equation, all u_j are different.

- $1 \le v_j \le 100$.
- $1 \le c_i, p_i \le 10000$.

Sample input and output

Sample Input	Sample Output
2 100 3 2 0 20 1 15 10 1 2 1 1 2 2 1 3 1 2 1 3 2 100 3 2 1 3 1 1 4 1 0 10 3 1 1 3 3 1 2 2	Case #1: 330 Case #2: 121

Problem J. Worried School

Time limit: 1 second

You may already know that how the World Finals slots are distributed in EC sub-region. But you still need to keep reading the problem in case some rules are different.

There are totally G slots for EC sub-region. X slots will be distributed among five China regional sites and Y slots will be distributed to the EC-Final. Of course X and Y are non-negative integers and $X + Y = G$.

Here is how the X slots be distributed:

1. Slots are assigned to the Asia Regional sites from the first place, the second place, $\cdots$, last place.
2. For schools having the same place across the sites, the slots will be given in the order of the number of "effective teams" in the sites.
3. No school could be assigned a slot 2 times, which means the schools will be skipped if they already got a slot.

After X slots are distributed, the EC-Final ranklist from highest rank will be assigned Y slots for those schools that haven't got a slot yet.

Now here comes a sad story, as X and Y are not announced until the end of the last regional contest of that year, even later!!!

Teachers from a school are worried about the whether they can advance to WF whatever the X and Y is. Let's help them find out the results before the announcement of X and Y.

Input

The first line of the input gives the number of test cases, T. T test cases follow.

Each test case starts with a line consisting of 1 integer and 1 string, G representing the sum of X and Y and S representing the name of the worried school.

Next 5 lines each consists of 20 string representing the names of top 20 schools in each site. The sites are given in the order of the number of "effective teams" which means the first site has the largest number of "effective teams" and the last site has the smallest numebr of "effective teams". The last line consists of 20 strings representing the names of top 20 schools in EC-Final site. No school can appear more than once in each ranklist.

Output

For each test case, output one line containing "`Case #x: y`", where `x` is the test case number (starting from `1`) and `y` is "ADVANCED!" if every non-negative value X, Y will advance the school. Otherwise, output the smallest value of Y that makes the school sad.

Limits

- $1 \leq T \leq 200$.
- School names only consist of upper case characters 'A' - 'Z' and the length is at most 5.
- $1 \leq G \leq 20$.

Sample input and output

Sample Input	Sample Output
1 10 IJU UIV GEV LJTV UKV QLV TZTV AKOV TKUV GAV DVIL TDBV ILVTU AKV VTUD IJU IEV HVDBT YKUV ATUV TDOV TKUV UIV GEV AKV AKOV GAV DOV TZTV AVDD IEV LJTV CVQU HVDBT AKVU XIV TDVU OVEU OVBB KMV OFV QLV OCV TDVU COV EMVU TEV XIV VFTUD OVBB OFV DVHC ISCTU VTUD OVEU DTV HEVU TEOV TDV TDBV CKVU CVBB IJU QLV LDDLQ TZTV GEV GAV KMV OFV AVGF TXVTU VFTUD IEV OVEU OKV DVIL TEV XIV TDVU TKUV UIV DVIL VFTUD GEV ATUV AKV TZTV QLV TIV OVEU TKUV UKV IEV OKV CVQU COV OFOV CVBB TDVU IOV UIV TKUV CVBB AKV TZTV VFTUD UKV GEV QLV OVEU OVQU AKOV TDBV ATUV LDDLQ AKVU GAV SVD TDVU UPOHK	Case #1: 4

Note

For the first test case, the optimal solution is $X = 6$ and $Y = 4$, at this time the advanced schools were [UIV, TKUV, QLV, CVBB, GEV, OCV, AKV, TZTV, VFTUD, UKV].

Problem K. Lazors

Time limit: 5 seconds

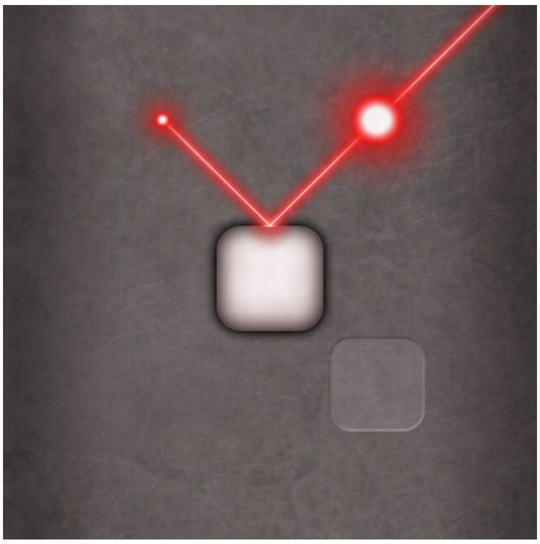
Lazors is a puzzle game of lazers and reflections. There is a board of size $N \times M$; some lazer transmitters and receivers installed on the board; and some tiles are initally placed in the board cells. The goal is to move the tiles so that all receivers get radiated by lazer beams with possible a series of reflections and refractions.

Transmitters are always at the center of one side of a cell. The direction of an emitted lazer beam is either diagnal or anti-diagonal. Each transmitter can be described by 4 parameters: `x`, `y`, `from_dir`, `to_dir`, where `from_dir` and `to_dir` is one of `N`, `S`, `E`, or `W`, representing north, south, east or west side of the cell. The first two parameters indicate the emitted lazer beam comes from cell on row `x`, column `y`; the last two indicate the direction of the emitted lazer beam.

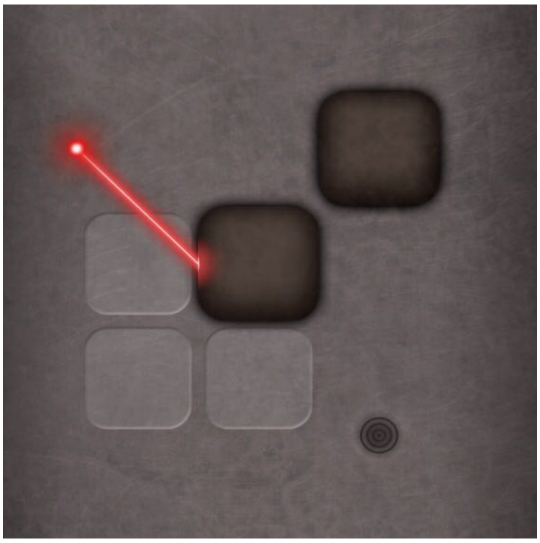
Receivers are also at the center of one side of a cell. Each receiver can be described by 3 parameters: `x`, `y`, `dir`, where `dir` is one of `N`, `S`, `E`, or `W`, representing north, south, east or west side of the cell. The parameters indicate the receiver is located at the center of `dir` side of the cell on row `x`, column `y`.

There are 4 types of tiles: white tile, black tile, glass tile, and prism tile. Each type of tile has its unique nature on lazer beam reflection and refraction.

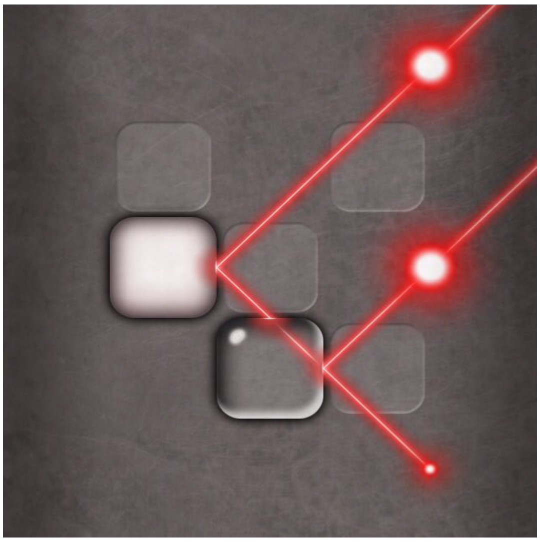
A *white* tile reflects lazer beams by law of reflection. In the following picture, the transmitter is the small red dot in the north-west corner, and the radiated receiver is the big red dot in the north-east corer.



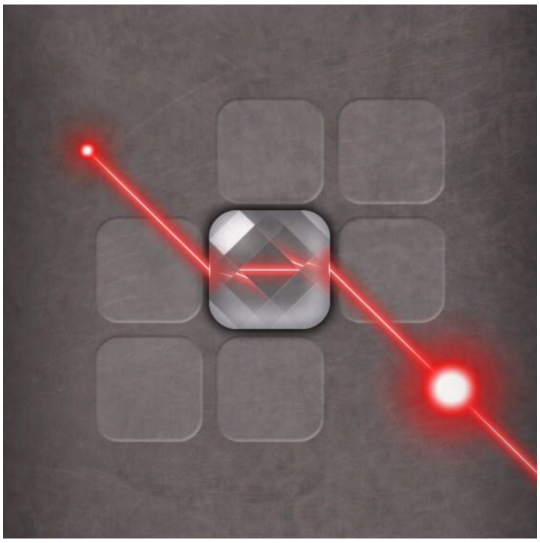
A *black* tile absorbs lazer beams. Lazer beams are stopped upon hitting the black tile. In the following picture, the transmitter is the small red dot in the north-west corner, and the un-radiated receiver is the target icon in the south-east corner.



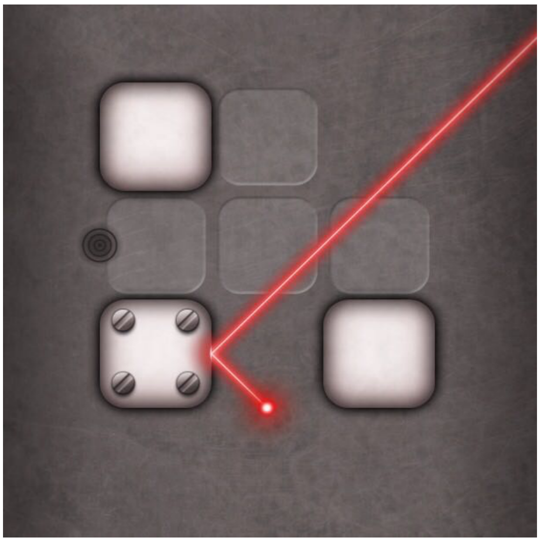
A *glass* tile split one lazer beam into two lazer beams, with one beam reflected by law of reflection and the other passing through the tile. In the following picture, the transmitter is the small red dot in the south-east corner, and the radiated receivers are the two big red dots above the receiver.



A *prism* tile refracts lazer beams. A lazer beam is "shifted" by one unit of grid and keeps going with the orignal direction. In the following picture, the transmitter is the small red dot in the north-west corner, and the radiated receiver is the big red dot in the south-east corner.



A *fixed* tile is locked on the board. You cannot move a fixed tile. In the following picutre, the white tile on row 2 column 2 is fixed. Black tiles, glass tiles, and prism tiles can also be fixed.



Input

The first line of the input gives the number of test cases, T. T test cases follow.

Each test case starts with a line containing two integers: N, M. Each of the following N lines contains M characters:

- '`.`' indicates an empty cell.
- '`#`' indicates an cell that forbids placing any tiles.
- '`o`' or '`O`' indicates a *white* tile.
- '`b`' or '`B`' indicates a *black* tile.
- '`x`' or '`X`' indicates a *glass* tile.

- 'm' or 'M' indicates a *prism* tile.
- Capitial letters indicate *fixed* tiles.

Next line contains an integer X, the number of transmitters. Each of the following X lines contains 2 integers `x`, `y`; and 2 characters `from_dir`, `to_dir`.

Next line contains an integer Y, the number of receivers. Each of the following Y lines contains 2 integers `x`, `y`; and 1 character `dir`.

Output

For each test case, output a line containing "`Case #x:`", where `x` is the test case number (starting from `1`). Next print the borad configuration with the same format as the input if there exists a solution, or print a line "`No solution!`". If there exist multiple solutions, any solution is accepted.

Limits

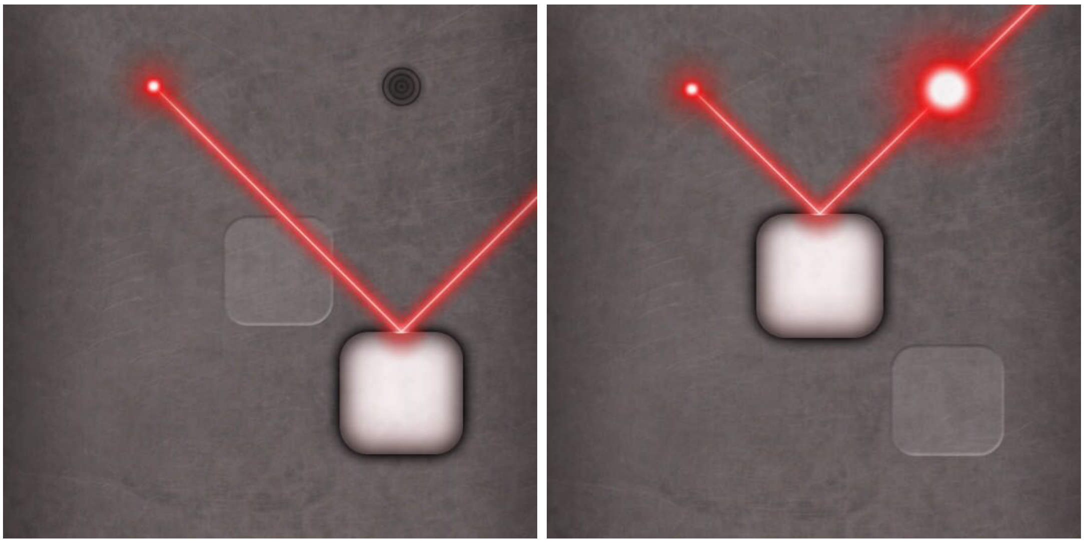
- $1 \leq T \leq 120$.
- $1 \leq N, M \leq 6$.
- $1 \leq N \times M \leq 24$.
- $0 \leq x < N$.
- $0 \leq y < M$.
- `dir`, `from_dir`, `to_dir` is one of `N`, `S`, `E`, or `W`.
- $1 \leq X \leq 2$.
- $1 \leq Y \leq 5$.

Sample input and output

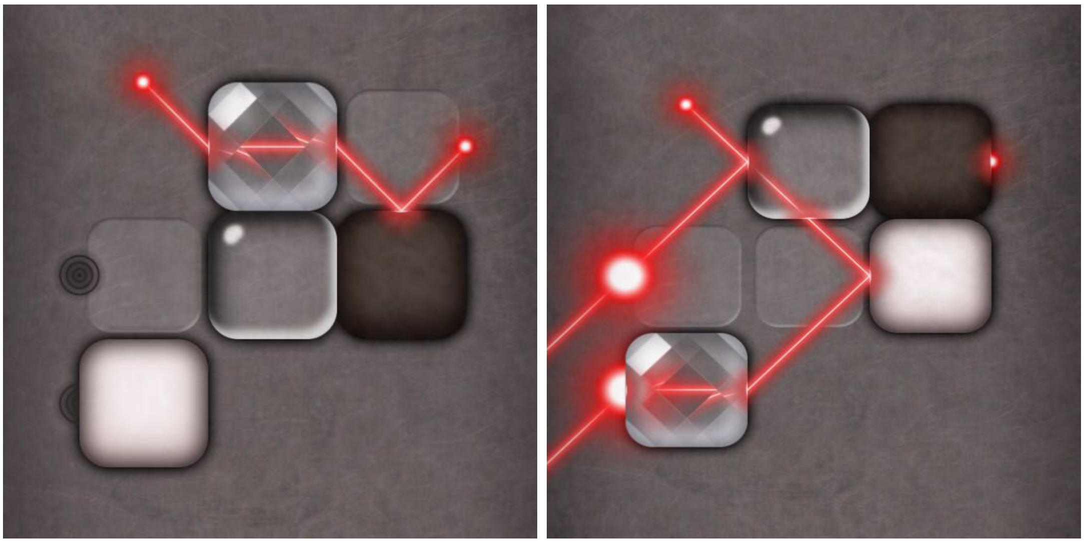
Sample Input	Sample Output
3 3 3 ### #.# ##o 1 0 0 N E 1 0 2 N 3 3 #m. .xb o## 2 0 0 N E 0 2 E S 2 1 0 W 2 0 W 4 4 o.bo obBo b.b. Bb.. 1 0 1 N E 1 2 3 E	Case #1: ### #o# ##. Case #2: #xb ..o m## Case #3: b.ob o.Bo b... Bbob

Note

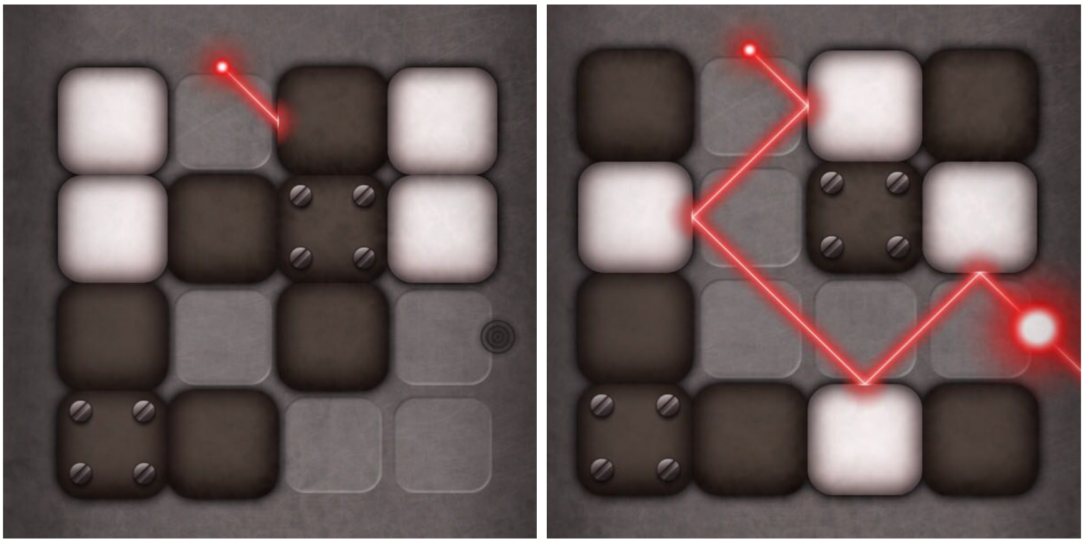
Case #1:



Case #2:



Case #3:



Problem L. Daylight Saving Time

Time limit: 1 second

Last month, xiaodao together with her friend poteko took a flight from San Francisco to Shanghai. When they were driving to the airport, xiaodao suddenly realized that the clock time on potekos car is one hour faster than the clock time on her mobile phone. And both of them might be correct, but how can it be? Because that day was Nov 6^{th}, the Daylight saving time switch day this year.

Daylight saving time (DST) or summer time is the practice of advancing clocks during summer months by one hour so that evening daylight lasts an hour longer. It is arguable that using DST can reduce overall energy consumption. Not all of us are using DST now, and for those regions adopting DST, the practices are also different.

In the case of California, effective in the U.S. in 2007 as a result of the Energy Policy Act of 2005, the local time changes from Pacific Standard Time (PST) to Pacific Daylight Time (PDT) at `02:00` to `03:00` on the **second Sunday in March** and the local time changes back from PDT to PST at `02:00` to `01:00` on the **first Sunday in November**.

Because it is one hour longer on that day, so xiaodao and poteko didn't miss the flights, but they found it still a little confusing. Interestingly, once xiaodao went back to Shanghai, she met a bug caused by exactly the same issue during the work. You might also be in trouble with DST some day, so here comes this problem and hope it will be helpful.

The local time in California without specifying whether it is PST or PDT could be ambiguous in some cases (e.g. `2016-11-06 01:25:00`). In this problem, you are given a local time in California. Check whether it is "`PST`", "`PDT`", "`Both`" or "`Neither`".

Input

The first line of the input gives the number of test cases, T.

T test cases follow. Each test case consists of one line, a date string written in the following format: `YYYY-MM-DD HH:MM:SS`.

Output

For each test case, first output one line containing "`Case #x: `", where `x` is the test case number (starting from `1`), following a result string which in one of "`PST`", "`PDT`", "`Both`" or "`Neither`".

Limits

- $1 \le T \le 1000$.
- date will be legal and between "`2007-01-01 00:00:00`" and "`2100-12-31 23:59:59`".

Sample input and output

Sample Input	Sample Output
4 2016-03-13 01:59:59 2016-03-13 02:00:00 2016-11-06 00:59:59 2016-11-06 01:00:00	Case #1: PST Case #2: Neither Case #3: PDT Case #4: Both